



Adaptive Threshold Wave Front Sensing for Portable Adaptive Optics System Driven by Hybrid Parallel Technique

Tianyu Zhang^{1,2,3,4} and Dongxia Wang¹

¹ School of Automation, Nanjing Institute of Technology, Nanjing 210000, China; 648008507@qq.com

² National Astronomical Observatories/Nanjing Institute of Astronomical Optics and Technology, CAS, Nanjing 210042, China

³ CAS Key Laboratory of Astronomical Optics and Technology, Nanjing Institute of Astronomical Optics and Technology, Nanjing 210042, China

⁴ University of Chinese Academy of Sciences, Beijing 100049, China

Received 2024 July 25; revised 2024 September 20; accepted 2024 October 10; published 2024 November 5

Abstract

Our Portable Adaptive Optics (PAO) system designed for high-contrast imaging of exoplanets with current 2–4 m class telescopes achieves a correction speed of nearly 1000 Hz, utilizing a Shack–Hartmann Wave Front Sensor (WFS) in a 9×9 sub-aperture configuration. As we look towards adapting the PAO system for larger telescopes, an increase in the number of sub-apertures in the WFS and enhanced precision in wave front detection are imperative. Originally programmed in LabVIEW, our initial PAO software is based on a traditional centroid calculation module for nighttime wave front sensing and lacks adaptive processing of background noise. To address these limitations and to boost the PAO system’s performance and accuracy in wave front detection, we propose a compressive neural network (Th-Net) combined with a specialized hybrid parallel programming approach for wave front detection. Our experimental results indicate that this hybrid parallel technique and Th-Net significantly enhance the PAO system’s operational speed and wave front detection precision under uneven background noise. This work paves the way so that a duplicable and low-cost PAO system can be used for direct imaging of exoplanets with large telescopes.

Key words: instrumentation: adaptive optics – instrumentation: high angular resolution – techniques: image processing

1. Introduction

Nighttime adaptive optics (AO) systems have significantly advanced over the past few decades, primarily for correcting wave front errors caused by atmospheric turbulence. Achieving diffraction-limited imaging performance in such systems necessitates rapid and accurate wave front error detection (Currie et al. 2023). Our Portable Adaptive Optics (PAO) system (Zhu et al. 2021), an evolution from the earlier 2–4 m class telescopes we utilized, is poised to operate as a visitor instrument on larger 8–10 m class telescopes (Marois et al. 2010). Consequently, the PAO’s Shack–Hartmann Wave Front Sensor (S-H WFS) requires an increased number of sub-apertures, potentially expanding from an 11×11 to a 29×29 configuration. This upgrade demands enhanced operational speed and more precise centroid calculation than the prototype.

AO systems commonly employ two types of hardware platforms: dedicated and universal (Rimmele 2004). The dedicated platform (Rao et al. 2010), used in systems like the Very Large Telescope (Ramsay et al. 2020) and the European Southern Observatory (ESO), relies on Field Programmable Gate Arrays and Digital Signal Processors for fast, low-latency digital image processing. While effective, they are costly and offer limited flexibility for different telescopes, with software

programming intricately tied to the hardware. Conversely, universal platforms, adopted by the Thirty Meter Telescope (Kahanamoku et al. 2020) and the European Extremely Large Telescope (Ramsay et al. 2020), utilize a commercial computer equipped with a graphics processing unit (GPU), offering easier access to software resources but presenting challenges in terms of hardware cost and future development. However, the cost of hardware and future development makes GPU a compelling choice for our PAO. For our PAO, balancing cost-effectiveness and performance suggests a preference for CPU-based systems (Schmidt et al. 2019). In previous work, we evolved our Portable Solar Adaptive Optics setup (Ren & Wang 2020) into a nighttime PAO system by substituting the daytime cross-correlation module with a centroid-calculation module. The PAO, leveraging LabVIEW programming and a commercial multi-core computer, comprises off-the-shelf optical and electronic components. LabVIEW, introduced by National Instruments, facilitates real-time control and programming of electronic hardware devices. It supports various hardware used in AO systems, like tip-tilt mirror (TTM), deformable mirror (DM), and wave front sensor (WFS), offering an extensive library of drivers and example codes. LabVIEW’s compatibility with a custom dynamic link library (DLL) enhances programming

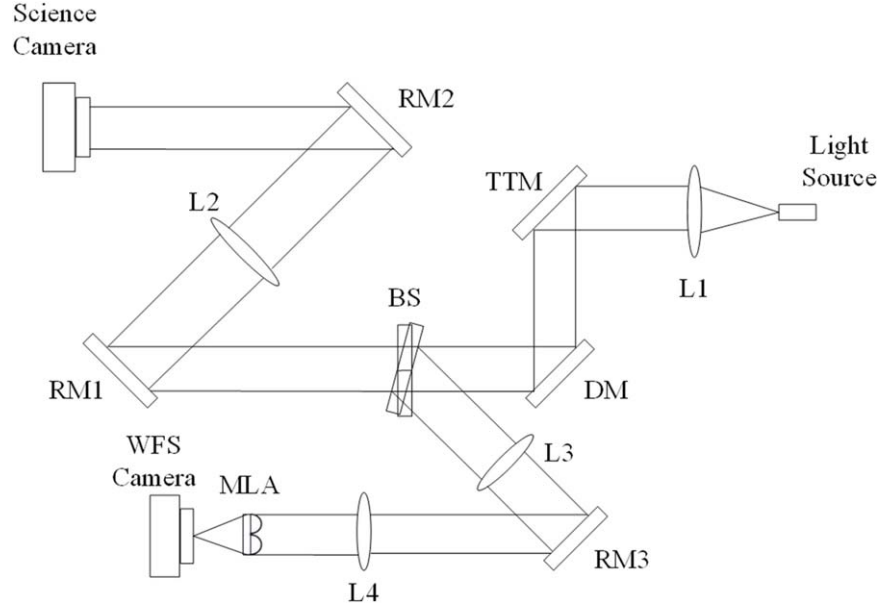


Figure 1. General principle of our PAO system.

productivity and reduces development time. Our PAO system is compact, measuring $900 \text{ mm} \times 780 \text{ mm} \times 300 \text{ mm}$, enabling rapid, cost-effective AO system deployment for smaller telescopes. Its efficacy was demonstrated in a stellar test observation on the 1.6 m McMath-Pierce Solar Telescope (Keller et al. 2003) at the National Solar Observatory, capturing clear images of white dwarf Sirius B in a binary system (Ren et al. 2009). The system's compactness and replicability also allowed successful application as a visiting instrument at the ESO 3.58 m New Technology Telescope (NTT).

In recent years, deep learning has been extensively applied in AO systems, particularly for S-H WFSs. Swanson et al. (2018) employed U-Net and long short-term memory networks to reconstruct wave fronts from a series of slope measurements. Jia et al. (2021) introduced a compressive S-H WFS utilizing deep neural networks, where only spot images with high signal-to-noise ratios (SNRs) were used for wave front reconstruction. DuBose et al. (2020) trained a deep neural network capable of mapping slope measurements and intensity measurements to original wave front difference maps. Guo et al. (2022) proposed an S-H Convolutional Neural Network (SH-CNN) to reconstruct Zernike modal coefficients from images captured by the WFS. These end-to-end algorithms, while powerful, are extremely computationally intensive and challenging to deploy on visiting instruments like our PAO. Even with high-performance and costly graphics workstations, it is difficult to achieve the necessary AO operation speeds required for real scientific observations.

Our PAO, optimized for 2–4 m class telescopes for high-contrast exoplanet imaging, employs a traditional centroid-

calculation module within its LabVIEW programming environment. However, this module is susceptible to various types of noise. We propose a new compressive neural network, named Th-Net, for wave front sensing in nighttime AO. Th-Net can generate adaptive thresholds for each sub-aperture in the S-H WFS across different SNR levels, effectively minimizing the impact of background noise. Despite its advantages, the processing requirements of Th-Net can slow down the overall system performance. Additionally, the original LabVIEW program used by the PAO lacks support for multi-core parallel processing, which limits the correction speed when adapting to larger telescopes.

To overcome these limitations, we have implemented a hybrid parallel programming technique. This approach integrates Th-Net with multi-processing parallel techniques (MPI/OpenMP) in C++ and Python programming, improving the accuracy and speed of centroid shift calculations in wave front aberration measurement. This method can be integrated as a sub-function of LabVIEW code, allowing the PAO system to maintain LabVIEW's ease in AO hardware control and software development while enhancing wave front sensing performance.

2. Initial PAO System and Performance

The first PAO system, established in 2013, was designed for high-contrast imaging of exoplanets using current middle-class telescopes with apertures of 2–4 m. Figure 1 illustrates the PAO system's general principle. A single-mode fiber, positioned on the telescope's focal plane, aids in testing and calibrating the system. The light emitted from this fiber, functioning as an

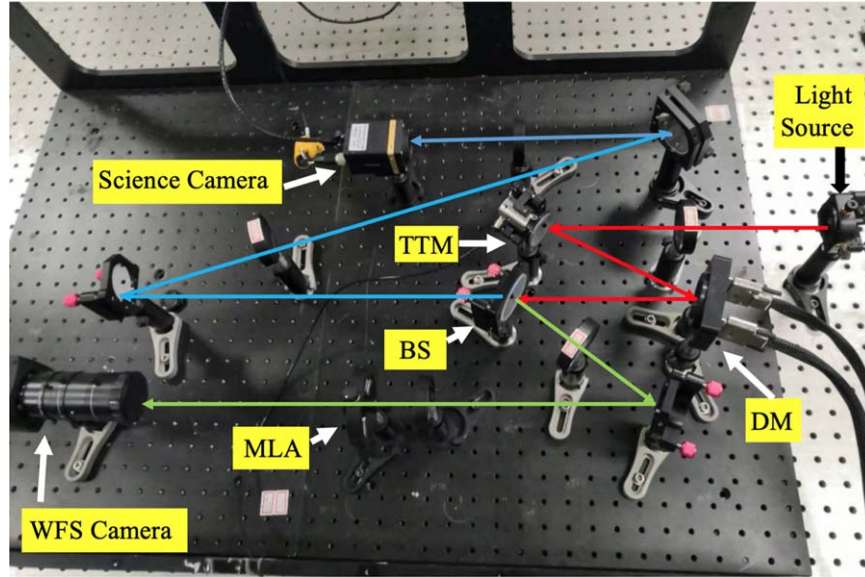


Figure 2. Photograph of the Compact PAO System.

ideal point light source, is collimated by lens L1, then reflected by a TTM and a DM. A beam splitter divides the light between the WFS path and the science imaging path. The latter includes lens L2 and Science Camera, while a combination of optical components L3, L4, and a microlens array works alongside the WFS camera for wave front sensing. Reflection Mirror (RM)1, RM2 and RM3 are used to redirect the light path. As a visiting system, the PAO is designed for minimal optical component usage, with only lenses L1, L2, L3, and L4 requiring replacement to adapt to different telescopes, ensuring its replicability across current middle-class telescopes.

Our initial PAO system employed a commercial multi-core computer for real-time control, managing tasks from receiving wave front image frames to executing corrective actions via electronic mechanisms like the TTM and DM. The system's software, written in LabVIEW, greatly simplifies programming and hardware control. Additionally, the LabVIEW code is flexibly programmed to accommodate different telescopes by allowing various S-H WFS configurations, such as 8×8 or 10×10 sub-apertures. This adaptability makes our PAO system suitable for integration with larger telescopes in the future. As all optical components, including lenses and mirrors, are off-the-shelf, we can rapidly duplicate an AO system for a specific telescope at a low cost. However, the quality of these commercial components, along with optical alignment errors incurred during the short setup time for real observations, can impact the system's diffraction-limited performance. To ensure quality imaging, a dedicated focal plane point-spread function (PSF) copy approach (Ren et al. 2021) is employed to remove these static aberrations. For easy integration with different telescopes, the entire PAO system is built into a compact size,

as depicted in Figure 2. Excluding the computer, all components are housed in a small enclosure measuring approximately $900 \text{ mm} \times 780 \text{ mm} \times 300 \text{ mm}$. This compact design facilitates the transportation of the PAO as a visiting instrument for scientific observations.

In July 2014, the ESO allocated us six nights on the NTT for engineering observations. However, due to strong winds at the site, only three nights were usable. Despite challenging conditions, our PAO, equipped with a 97-actuator DM from Alpao and a 9×9 sub-aperture S-HWFS, successfully locked onto an 8th magnitude star for wave front sensing. Typically, the PAO system achieves a correction speed of nearly 1000 Hz with an i7 8-core CPU operating at 3 GHz. These engineering observations demonstrated that the PAO could be feasibly connected to a middle-class telescope without performance degradation. Post-operation with the NTT, the PAO was upgraded with a new 97-actuator DM capable of operating at 2000 Hz. Further tests at the 3.5 m Astrophysical Research Consortium telescope in the Apache Point Observatory proved successful. The S-H WFS configuration remained at 9×9 sub-apertures, sampled by 4×4 pixel imaging, achieving a contrast of 1.3×10^{-5} at an inner working angle of $0''.36$. With longer exposure times (2 hr), a contrast of 3.3×10^{-6} was attainable (Zhu et al. 2021). To fully exploit the PAO's capabilities, the static aberration in PAO was effectively corrected by measuring the difference between system PSF and perfect reference PSF, and directly command the DM to copy the perfect reference PSF into the system to remove all the static aberrations from the AO system (Ren et al. 2021). This technique allows the use of low-cost, commercial optical components in building our PAO. As all static aberrations are

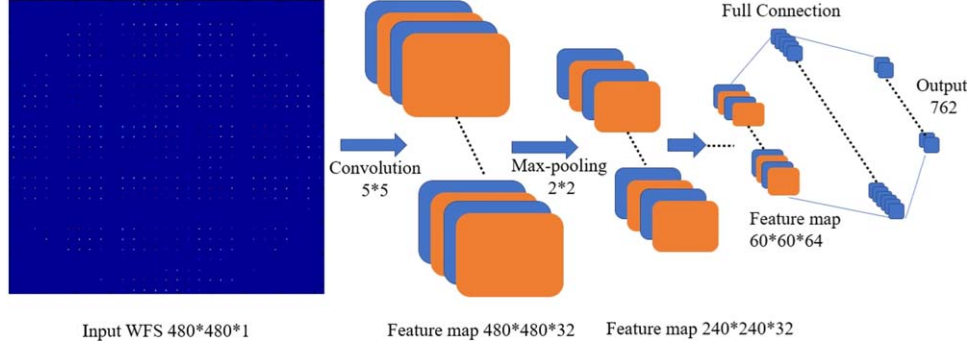


Figure 3. Sketch map of threshold generation based on Th-Net.

correctable through our dedicated PSF copy approach, only the dominant wave front errors need to be considered for evaluating the PAO's performance. The residual variance of our AO system with S-H WFS can thus be calculated as the remaining error.

$$\sigma^2 = \sigma_{\text{fit}}^2 + \sigma_r^2 + \sigma_{\text{ph}}^2 + \sigma_{\text{bw}}^2, \quad (1)$$

where σ_{fit}^2 is the fitting error that represents the wave front variance due to the number of DM actuators, which is limited compared to the telescope aperture. σ_r^2 is the WFS read noise error, σ_{ph}^2 is the photon noise error and σ_{bw}^2 is the lag error introduced by the finite bandwidth. For our PAO, the fitting error is determined by the number of DM actuators. If a DM sample has spacing r_s and seeing r_0 , it should be calculated as $\sigma_{\text{fit}}^2 = 0.349(r_s/r_0)^{5/3}$. $\sigma_{\text{bw}}^2 = (f_G/f)^{5/3}$, where $f_G = 0.43(v/r_0)$, and v is the wind speed (Tyson & Frazier 2022). For small residual variance, the Strehl ratio (SR) can be calculated as

$$\text{SR} = e^{-\sigma^2}. \quad (2)$$

In fact, the AO performance is relevant to the above error source as well as the natural guide star (NGS) brightness, seeing condition, telescope aperture size, sub-aperture number of S-H WFS, DM/S-H WFS's geometric configuration and other properties. In our engineering observation, under the good seeing $r_0 = 13$ cm at 500 nm, an SR of 0.64 can be reached at the H (1.6 μm) band, for an NGS with star magnitude of 5.

3. Centroid Shift Measurement

In the original configuration of our PAO system, we utilized the traditional Center of Gravity (COG) algorithm for measuring centroid shifts. This process involves calculating the centroid of a spot in each sub-aperture and determining the centroid shift relative to the ideal spot coordinates, thereby computing the wave front slope. While the traditional COG algorithm is advantageous due to its low computational requirements, it is notably susceptible to noise during nighttime observations, especially when signal strength is weak.

To mitigate this issue, thresholding emerges as a crucial processing algorithm in wave front sensing, effectively reducing unwanted background noise and random fluctuations. The implementation of this algorithm involves applying a threshold (T) to each pixel P'_{ij} located at row i and column j . The specific formula and methodology for this threshold application are detailed in Equation (3)

$$P'_{ij} = \begin{cases} P_{ij} - T & \text{if } P_{ij} \geq T \\ 0 & \text{if } P_{ij} < T \end{cases} \quad (3)$$

In our earlier PAO system, the strategy for threshold selection involved conducting noise statistics at the four corners of the target surface and selecting a uniform threshold value for the entire target area. However, our PAO system will be adapted for large-aperture ground-based telescopes in the future. As the number of sub-apertures in the S-H WFS will significantly increase, and the intensity of spots across different sub-apertures on the target surface will change somewhat with the variable background scenarios, a uniform threshold fails to account for different application scenarios. This leads to some sub-apertures having thresholds that are too high, while others have thresholds that are too low, preventing the optimal threshold from being achieved. To address this issue, we proposed a compressive neural network named Th-Net for adaptive threshold generation of each sub-aperture in an S-H WFS in this paper. The Th-Net can intelligently process the entire image and output individual thresholds for each sub-aperture simultaneously, thereby minimizing the impact of noise.

3.1. Adaptive Threshold Generation Architecture

The principle of the adaptive threshold generation architecture is illustrated in Figure 3. The input images, sized 480×480 , are sampled by the S-H WFS. The first layer of our Th-Net is a convolutional layer with a 10×10 kernel size and 32 channels, resulting in a feature map of $480 \times 480 \times 32$. This feature map is subsequently sub-sampled by a max-

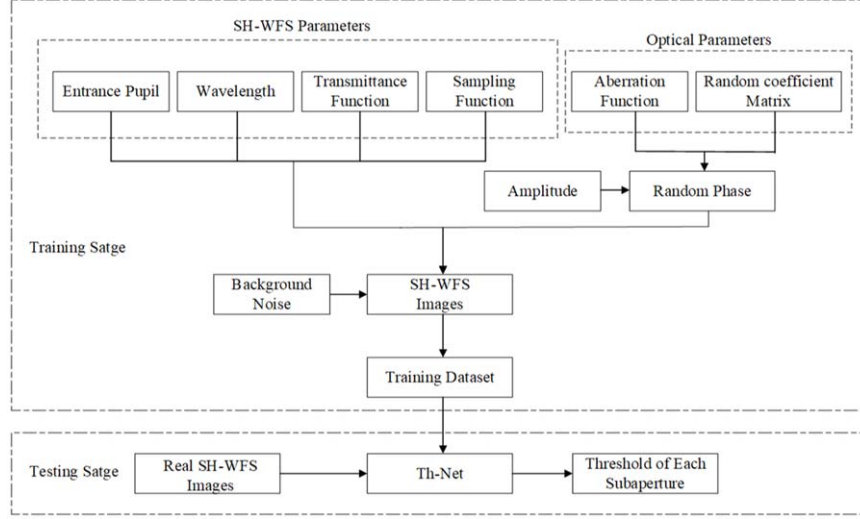


Figure 4. Data set generation procedure.

pooling layer with a 4×4 window and a stride of 2, reducing the spatial dimensions while retaining the most salient features. Detailed parameters of the full Th-Net are provided in Table 1. The feature map, now reduced to $29 \times 29 \times 64$ after the final max-pooling layer, is flattened and fully connected to a 100-dimensional dense layer. The adaptive thresholds for each sub-aperture are determined by the final fully connected layer, which has 762 nodes corresponding to the number of sub-apertures in the S-H WFS used in the AO system.

For relieving the computational burden for better real-time performance, the architecture of the Th-Net is designed to be as compact as possible. This Th-Net, which only retains enough layers for threshold detection, can greatly reduce the parameters needed for training. As a result, the performance requirements for hardware can be significantly reduced when deploying Th-Net in AO system for future large telescope.

3.2. Data Set Generation and Training

In real scientific observations, our PAO system, as a visiting instrument, is subjected to a wide range of harsh environmental conditions. A robust training data set is crucial to handle diverse observational scenarios. Consequently, synthetic data, generated based on Fourier optics, are employed for proof-of-concept and experimental validation. The data set generation procedure is depicted in Figure 4. Essential parameters such as the wavelength, entrance pupil function, sampling function, and lens transmittance function must be specified for the S-H WFS. The random coefficient matrix should be initialized randomly and adhere to Kolmogorov turbulence (Rodier 1990), given that the resulting data will be used for training AO wave front sensing algorithms. The random phase

Table 1
The Detailed Parameters of Full Th-Net

Type of Layer	Input Size	Filter	Stride	Pad	Kernels
Input	$480 \times 480 \times 1$
Convolution	$480 \times 480 \times 1$	10×10	1	2	32
ReLU	$480 \times 480 \times 32$
Pooling	$480 \times 480 \times 32$	4×4	2
Convolution	$240 \times 240 \times 32$	6×6	1	1	32
ReLU	$240 \times 240 \times 32$
Pooling	$240 \times 240 \times 32$	4×4	2
Convolution	$60 \times 60 \times 32$	6×6	1	...	64
ReLU	$120 \times 120 \times 64$
Pooling	$120 \times 120 \times 64$	4×4	2
Fully connected	1×57600
Fully connected	1×1024

of the complex amplitude at the entrance pupil is defined as shown in Equation (4)

$$\phi_i(x_0, y_0) = \sum_{k=1}^K C_i(k) D_k(x_0, y_0), \quad (4)$$

where C_i is the i_{th} coefficient vector and $D_k(x_0, y_0)$ is the modal function like Zernike modes or Karhunen–Loève modes used to decompose the phase with a maximal order K . The complex amplitude on the entrance pupil is defined as in Equation (5)

$$U_i(x_0, y_0) = A(x_0, y_0) e^{i\phi_i(x_0, y_0)}, \quad (5)$$

where $A(x_0, y_0)$ is the amplitude that has to be fixed with a uniform, Gaussian or any other distribution that follows the real conditions. Following the principle of Fourier optics, the S-H

WFS images can be calculated as the following function (6)

$$I_i(m, n) = S(m, n, x_f, y_f) \left| \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} U_i(x_0, y_0) \right. \\ \times P(x_0, y_0) T(x_0, y_0) \exp \left[-i \frac{2\pi}{\lambda f} (x_f x_0, y_f y_0) \right] \\ \times dx_0 dy_0 \left. \right|^2 \quad (6)$$

where f is the focal length of the microlens, λ is the wavelength, $T(x_0, y_0)$ is the transmittance function of the lens, and $S(m, n, x_f, y_f)$ represents the sampling function we defined. In addition, the labels corresponding to each S-H WFS image are also indispensable in neural network training. Obviously, the label of each S-H WFS image is the threshold of each sub-aperture in our case. For getting enough label data, threshold (Th_{ij}) for a sub-aperture located at row i and column j is calculated using Equation (7)

$$Th_{ij} = a_{ij} + gb_{ij}, \quad (7)$$

where a_{ij} is the noise mean in current sub-aperture, b_{ij} is the according noise standard deviation, and g , the value range of which is $[0, 4]$, is the scale factor. In this paper, g is usually set as 3. The training data set was created under seeing conditions where r_0 averages 13 cm, and the scientific target is assumed to be at an infinite distance, approximated as a point source. The input light amplitude over the entrance pupil was varied between 0.4 and 1 to simulate realistic observation conditions, enhancing the robustness and adaptability of the Th-Net. A total of 50,000 samples of S-H WFS images, along with corresponding thresholds for each sub-aperture, were generated. This data set includes a training set of 40,000 samples, a validation set of 5000 samples, and a test set of 5000 samples. An example is illustrated in Figure 5. The architecture of the SH-CNN described in the figure and table has an input of 480×480 D (S-H WFS images) and output of 762D (threshold of each valid sub-aperture). The training of the Th-Net was performed on a laptop with a GPU (NVIDIA GTX 3060) using TensorFlow (Gulli et al. 2019). The Adam optimizer with the initial learning rate of 0.0002 was used. The time required for 50 epochs of the whole training was about 14 hr. The training result is shown in Table 2.

Employing this Th-Net allows each sub-aperture in our PAO system to generate its local optimal threshold based on the actual noise and signal levels observed in each calibration cycle. However, this preprocessing step tends to considerably slow down the measurement speed of the spot centroids, a challenge that becomes more pronounced as the number of sub-apertures in the WFS increases. This issue is especially evident when the sub-aperture configuration expands significantly, such as from 9×9 to 29×29 . Furthermore, the original LabVIEW implementation of our COG algorithm does not effectively leverage multi-core parallel computing capabilities.

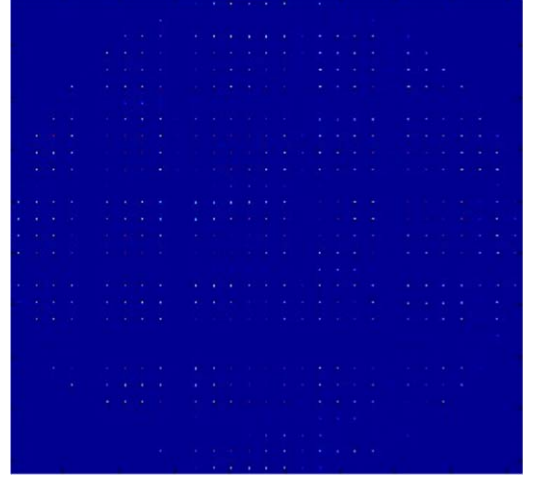


Figure 5. The sample S-H WFS image generated by Th-Net.

Table 2
The Training Result of Th-Net

Network	Train Loss	Val Loss	Test Loss
Th-Net	0.00036	0.00035	0.00032

A significant limitation hindering the application of our PAO system with larger telescopes is the increased demand for rapid data processing. Even with the deployment of deep neural networks such as UNet for wave front reconstruction, which eliminates the need for slope calculation, the entire process from wave front reconstruction to control command generation remains insufficiently fast in practice. Therefore, this paper introduces a parallel hybrid programming control technique designed to work in conjunction with Th-Net, addressing the bottleneck in overall processing speed.

4. Hybrid PAO Programming

To address the challenges of correction speed attenuation due to an increased number of sub-apertures and the introduction of Th-Net, and to surmount the limitations of the original LabVIEW COG codes in parallel computation, we propose a hybrid approach. This approach integrates a LabVIEW block-diagram based program with advanced parallel techniques using MPI and OpenMP, enhancing our PAO system. Figure 6 illustrates the controlling diagram of this upgraded system.

The parallel computation is executed through C++ and Python code, employing shared memory technology (Bershad et al. 1991). For efficient data transmission and processing, OpenMP/MPI parallel code is integrated into our original PAO LabVIEW code via a LabVIEW DLL. Additionally, the shared

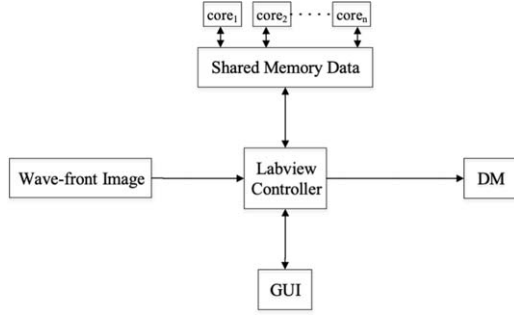


Figure 6. Schematic diagram of the upgraded PAO control structure.

memory in our control software facilitates high-speed data operations by employing multi-cores, offering significant advantages in data transfer speeds. The C++ and Python codes are designed to read data of wave front images and relay spot centroid shift calculations back to the LabVIEW code via the shared memory DLL. This section will demonstrate that our hybrid programming technique effectively utilizes CPU resources for multi-core parallel computation at a low cost. The commercial computer used for testing this hybrid parallel technique is equipped with two Intel Xeon E5-2680 CPUs, each with 12 cores operating at 3 GHz. The selection of core numbers is crucial for the high-speed measurement of parallel spot centroid shifts, with a few cores reserved for other essential tasks like hardware control and operating system operations.

Our hybrid programming employs MPI (Gropp 2000) and OpenMP (Jin et al. 2011), two open-source Application Program Interfaces known for their excellent performance in parallel computation (Smith & Bull 2001). This technique operates hierarchically, with MPI parallelism at the upper level and OpenMP providing lightweight parallelism within each MPI process. To ensure effective data transfers between each MPI process, point-to-point block sending and receiving message functions, “MPI – Send” and “MPI – Recv” are utilized as a message-passing mechanism, also enforcing data synchronization. Conversely, communication between each OpenMP thread is realized through access to shared memory of each process. Given that the centroid shift of the spot in each sub-aperture must be measured and calculated individually, we developed a dedicated MPI/OpenMP parallel technique to fully support the parallel computation for our PAO WFS.

Figure 7 illustrates the parallel structure of MPI and OpenMP. In this configuration, MPI, positioned outside the OpenMP parallelism, is set in the upper layer for process scheduling. Within each MPI process, OpenMP is tasked with the generation and scheduling of threads. Suppose the commercial multi-core computer has N cores and there are M effective sub-apertures requiring processing. In that case, our hybrid parallel program initiates by loading pre-processing

calibration data, such as the perfect spot centroid coordinates of the WFS. The whole flowchart of our Th-Net combined with the hybrid parallel technique is shown in Figure 8. Following this, the workflow is executed according to the subsequent procedure:

- (1) Receive the focal images from S-H WFS;
- (2) Generate adaptive threshold of each sub-aperture through Th-Net;
- (3) Set number of processes by MPI as m , with each process containing M/m sub-apertures;
- (4) Allocate according to threshold and reference centroid coordinates to each process;
- (5) Set number of threads by OpenMP as N/m to parallelize the computation of centroid shift in each process and calculate centroid shift of M/N sub-aperture’s spot in each thread;
- (6) Collect computation results and calculate the wave front slope;
- (7) Compute the controlling signals by the direct slope method to DM.

5. Centroid Computation and Performance

The centroid shifts in our system are determined by calculating the difference between the centroid of each spot within a sub-aperture and its corresponding reference centroid. Consequently, the precision of these centroid shifts is contingent upon the accuracy with which the spot centroids in each sub-aperture are calculated. In this paper, we use the centroid calculation results (x_s, y_s) obtained under conditions of extremely high SNR as the baseline for computing the Centroid Offset Error (CEE), as delineated in Equations (8) and (9). It is important to note that (x_s, y_s) are utilized solely to assess the accuracy of centroid calculation, and the actual centroid shifts are derived from pre-calibrated reference centroids.

$$\Delta x_{\text{CEE}} = \sqrt{(x - x_s)^2} \quad (8)$$

$$\Delta y_{\text{CEE}} = \sqrt{(y - y_s)^2}. \quad (9)$$

In practical experiments, discerning a distinct boundary between the signal and noise of a spot in each sub-aperture proves challenging. Furthermore, estimates of the SNR can differ across various regions. Therefore, our focus is on effectively gauging the high or low state of SNR within each sub-aperture to select the standard position (x_s, y_s) for our measurements. This paper employs an SNR estimation method grounded in the Gaussian spot model, as follows

$$\text{SNR}_{ij} = \frac{I_{\text{max}} - a}{b}. \quad (10)$$

Here I_{max} is the maximum value of current sub-aperture light intensity before the threshold operation, a is the noise mean, and b is the noise standard deviation. In order to evaluate the

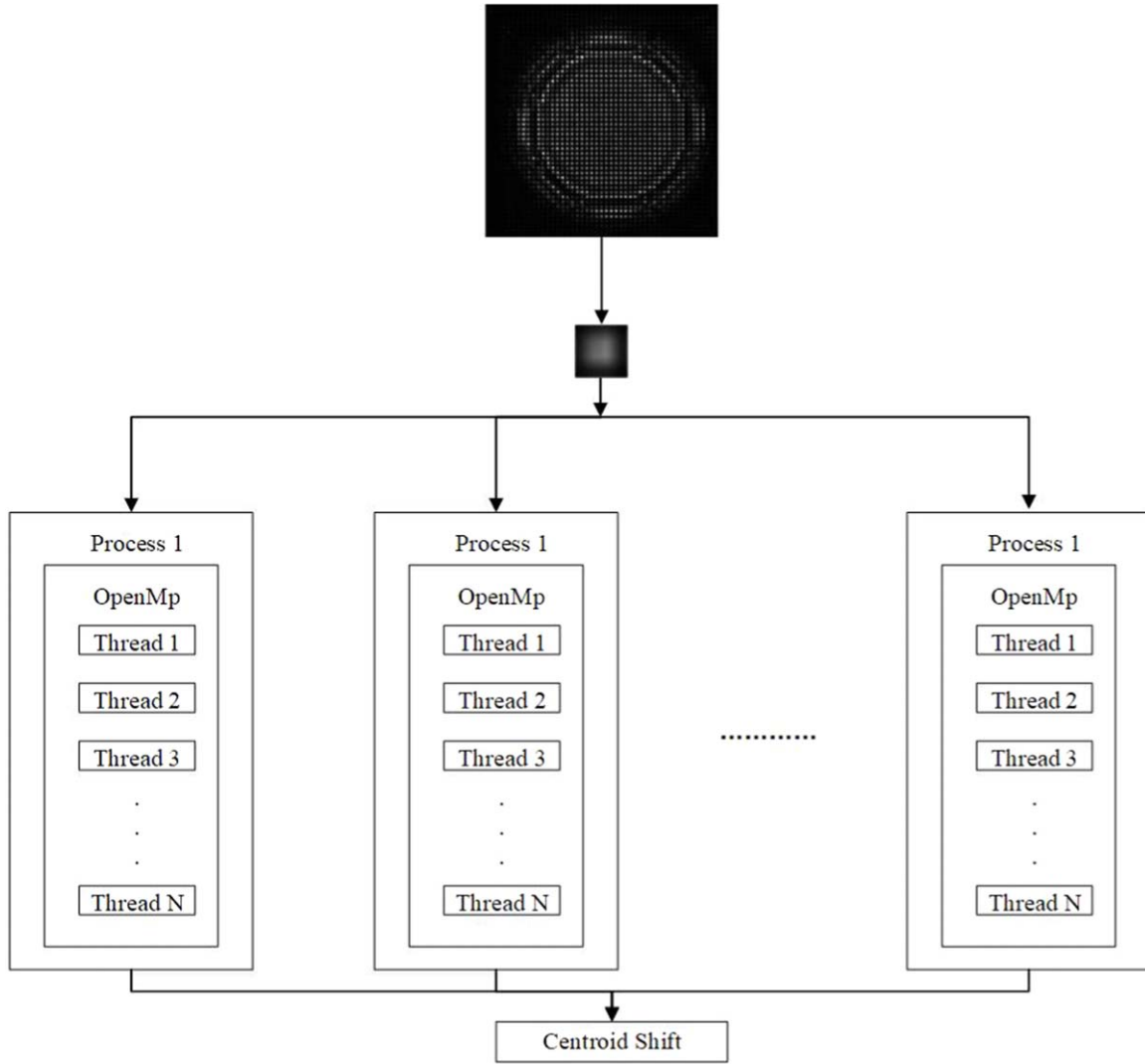


Figure 7. Parallel structure of MPI and OpenMP.

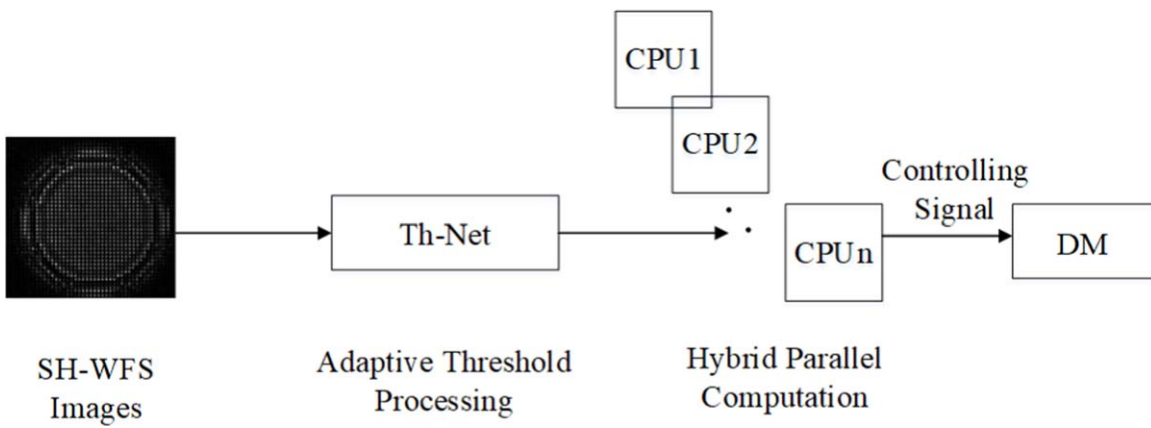


Figure 8. The whole flowchart of our Th-Net combined with the hybrid parallel technique.

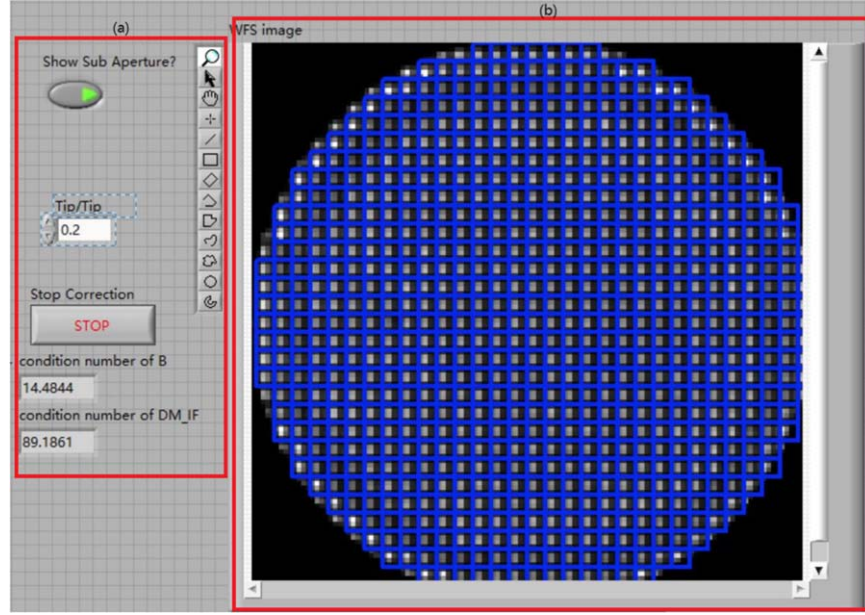


Figure 9. GUI of the centroid shift simulation software, in which there is a 29×29 WFS configuration.

performance of the parallel computation, a simulation software package based on LabVIEW is developed. As shown in Figure 9, the operation interface of simulation software consists of two parts. The parameter setting is shown in section (a); in left of section (b), the WFS image is displayed and the centroid shift result will be saved as a .txt file. The selected or effective sub-apertures are labeled with blue grids in the wave front measurement process. For simulating the real observation, a 29×29 WFS configuration is used in this evaluation. The traditional COG algorithm, implemented using a LabVIEW block-diagram based program, has been effectively utilized in our original PAO system for centroid shift measurement. This provides a baseline to compare with our new parallel technique in terms of speed and accuracy. For performance testing, we used a commercial multi-core computer (HP DESKTOP-6QUILPR) equipped with two Intel Xeon E5-2680 CPUs, each comprising 12 cores.

As previously discussed, the accuracy of the parallel algorithm is evaluated by calculating the CEE using Equations (6) and (7). In this paper, we first assessed the multi-frame SNR of each sub-aperture at different exposure times, selecting the centroid coordinates corresponding to the maximum SNR as the standard coordinates. Table 3 presents the various SNR levels. Subsequently, we processed the WFS image using both the traditional global threshold (T1) from the original PAO and the newly implemented adaptive threshold (T2) generated by Th-Net to determine the CEE. Table 4 illustrates the overall mean CEE across all effective sub-apertures when employing T1 and T2 at different exposure times. It is evident that the application of T2 significantly enhances the accuracy of centroid measurement,

Table 3
SNR at Different Exposure Times

Exposure Time (ms)	SNR
10	465
5	349
1	191

Table 4
Calculation Accuracy Results of T1 and T2 at Different Exposure Times

Exposure Time (ms)	10		5		1	
	Δx_{CEE}	Δy_{CEE}	Δx_{CEE}	Δy_{CEE}	Δx_{CEE}	Δy_{CEE}
T1	0.187	0.169	0.123	0.116	0.098	0.076
T2	0.039	0.021	0.048	0.029	0.077	0.059

achieving accuracies of 0.039 and 0.021 in the x - and y -directions, respectively. Consequently, the Th-Net is incorporated into our hybrid parallel algorithm. Please note that the seeing r_0 is averaged at 13 cm in our simulation or test and Th-Net will be first connected to the S-H WFS and run for 5–10 minutes to optimize the thresholds for the current environment before starting testing or observation.

The process of centroid calculation in each sub-aperture can be executed using a for loop in LabVIEW. In our previous work, we deployed LabVIEW's original parallel model by enabling parallel execution for loop iterations. This paper

Table 5

The Running Speed of Wave Front Measurement with Various Num-core

Process	Num-thread	Num-core	Speed (Hz)
2	4	4	3624
2	4	6	5180
2	4	8	5853
2	4	10	6010
2	4	12	4888
2	4	14	3971
2	4	16	3701
2	4	18	3656
2	4	20	3430

introduces a hybrid parallel technique to accelerate for loop execution. To optimize the performance of this technique, it is crucial to set the number of the desktop's CPU cores (Num-core), OpenMP threads (Num-thread), and MPI processes appropriately to maximize the utilization of computational resources. Various configuration schemes have been tested, and a model utilizing two MPI processes with four OpenMP threads was found to offer the best stability in long-term stress tests. To evaluate the running speed of our hybrid parallel technique, we implemented the technique (2 processes, 4 threads) with varying Num-core. As indicated in Table 5, the configuration of 2 processes, 4 threads, and 10 cores achieved the highest running speed of nearly 6000 Hz. As shown in Figure 10, the wave front measurement computation time fluctuates with the CPU core utilization. The minimum computation time is achieved when 10 cores are used, with the core utilization reaching 42%. It is important to note that although the desktop used has 24 cores, several cores must be reserved for the hardware system and other computing requirements. Compared to the traditional COG algorithm executed by a single LabVIEW block-diagram based program, which attains a running speed of 2800 Hz, our hybrid parallel technique has increased the calculation speed by 114% in centroid measurement.

Finally, we integrated this hybrid parallel technique software into our PAO system and measured the PAO operation speed at different WFS configurations of 11×11 , 20×20 and 29×29 . The results are detailed in Table 6. In these experiments, the iXon Ultra 888 camera produced by Andor, an EMCCD camera with frame rates of 26 fps at 1024×1024 pixels and 93 fps at 512×512 pixels, was used as the WFS camera. In our tests, the PAO system, augmented by the hybrid parallel technique, achieved operation speeds of 2116 Hz, 1119 Hz, and 511 Hz for 97, 316, and 762 effective WFS sub-apertures, respectively. It is evident that the PAO system, when supported by the hybrid parallel technique, attains faster running speeds than the LabVIEW block-diagram based program across different WFS configurations. However, it is worth noting that

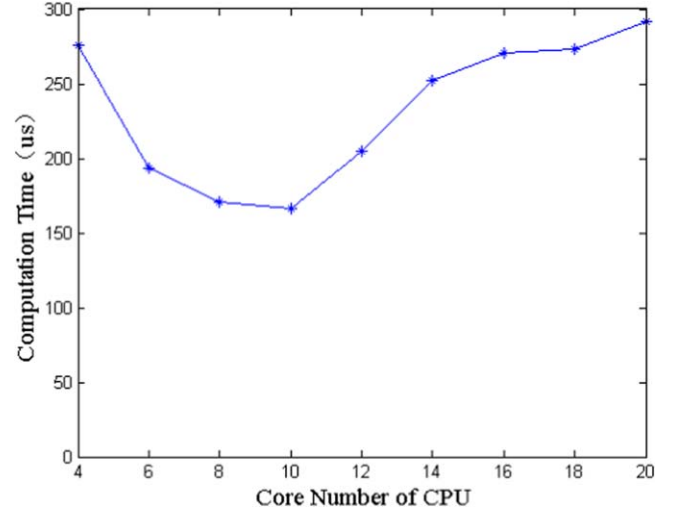


Figure 10. The computation time and CPU jitter of our wave front measurement driven by the hybrid parallel technique.

Table 6

PAO Running Speed Driven by Hybrid Technique and Single Mode in Different WFS Configurations

WFS Configuration	11×11	20×20	29×29
PAO speed driven by hybrid technique (Hz)	2116	1119	511
PAO speed driven by single LabVIEW (Hz)	989	634	298

the overall performance of the PAO system is somewhat constrained by the limited image capture speed of the WFS camera.

6. Laboratory Test and Result

Since our PAO system is specifically designed for nighttime observations, it employs a point light source to assess performance metrics such as the SR, a critical step in characterizing the performance of an AO system (Tyson & Frazier 2022). To accurately gauge the performance of our PAO system equipped with the hybrid parallel technique under realistic nighttime observation conditions, we have established a general platform for atmospheric turbulence perturbation in our laboratory. This platform can simulate turbulence perturbations corresponding to various wind speeds and seeing conditions. Theoretical models of atmospheric turbulence are crucial in evaluating the performance of nighttime AO systems. Traditionally, an atmospheric turbulence simulation utilizes a rotatable phase plate to create time-varying wave front aberrations. However, this method requires the fabrication of specialized phase plates with specific spatial frequency components, such as those described by the Kolmogorov

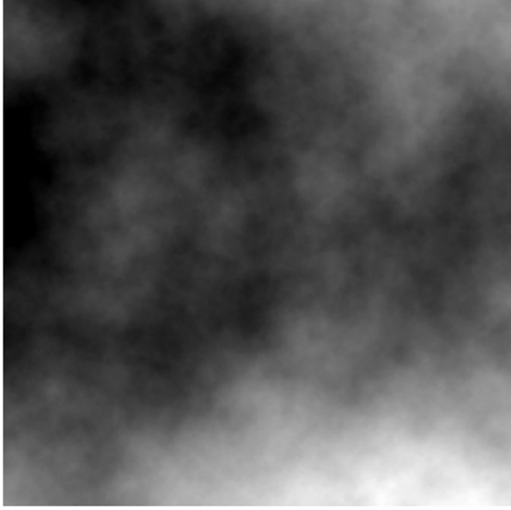


Figure 11. Turbulence phase screen.

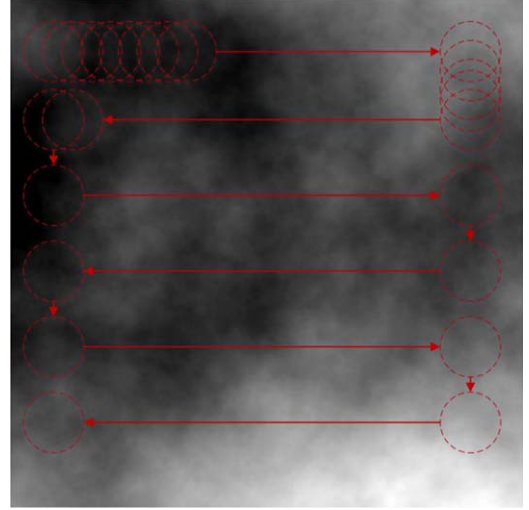


Figure 12. The simulation of telescope's moving path around the phase screen.

Power Spectrum, making it challenging to generate a truly random phase (Lee et al. 2012). To achieve dynamic wave front turbulence, liquid-crystal spatial light modulators are used in some simulators, but these can only provide moderate light transmission and require polarized light. Alternatively, a DM can also be employed in atmospheric turbulence simulation (Lee et al. 2017). In this paper, we use a DM to generate a phase screen at speeds exceeding 1000 Hz, which is more than sufficient for the performance evaluation of our PAO system.

As depicted in Figure 11, our atmospheric turbulence simulator employs a $10 \times 10 \text{ m}^2$ phase screen generated using the Object-Oriented Matlab Adaptive Optics (OOMAO) simulation tool (Conan & Correia 2014). To mimic real observational conditions, we utilized the OOMAO tool to create a phase screen of the same dimensions, with a Fried parameter of 13 cm, indicative of good nighttime seeing conditions. Figure 12 highlights a 1 m diameter telescope, represented by a red circle, traversing across the entire phase screen at different instants. The red path and arrow illustrate the telescope's movement, simulating actual telescope motion. In our experimental setup, the telescope samples the phase screen sequentially from left to right and then from top to bottom, completing a full cycle. This cycle is repeated in an iterative loop to simulate real observational conditions. It is important to note that the speed at which the telescope traverses the phase screen effectively simulates wind at a specific speed, achieved by adjusting the telescope's movement speed. Figure 13 presents the Graphical User Interface (GUI) of the simulator, which was developed using LabVIEW. The complete schematic of this simulator is shown in Figure 14. The setup involves light emitted from a point source being collimated by a lens, then reflected by a DM. In our simulator, the DM is capable of generating phase errors following the Kolmogorov

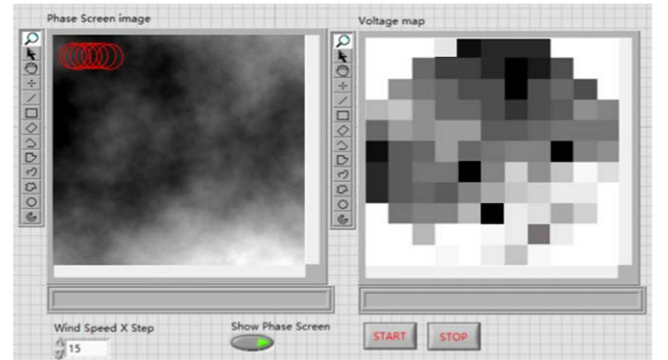


Figure 13. GUI of the atmosphere turbulence simulator.

Power Spectrum, controlled via LabVIEW code. Post-reflection from the DM, the light passes through an RM and another lens, culminating as the simulator's output. This output then serves as the input for testing our PAO system, allowing us to closely replicate and analyze actual observational conditions.

Our PAO system is established according to the schematic diagram that is shown in Figure 1. The DM, manufactured by the ALPAO Corporation, is equipped with 88 actuators in a 10×10 configuration. The TTM is sourced from PI Corporation (PI S-300.2SH). An HeNe laser light source of wavelength = 632.8 nm, which can be seen as a perfect point light source, is used in our test via single-mode fiber. For high-speed wave front sampling, the EoSens 3CXP camera developed by Mikrotrotron GmbH serves as the WFS camera. However, static aberrations inherent in our PAO system's light path, such as Non-Common Path Aberration and AO Optical Residual Static Aberration, will inevitably impact the system's imaging performance. Consequently, it is essential to effectively

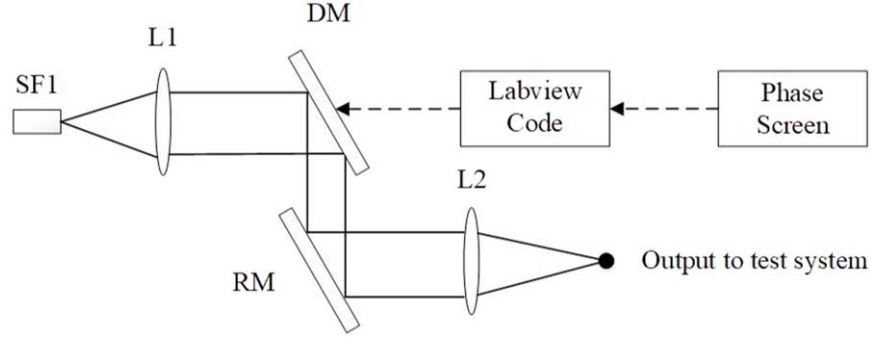


Figure 14. The complete schematic of atmospheric turbulence simulator.

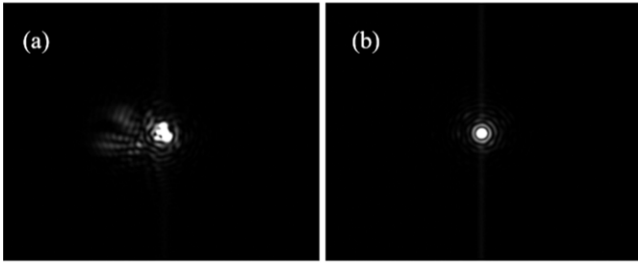


Figure 15. Focal plane PSF images without (a) and with (b) AO static aberration.

eliminate all static aberrations before integrating the PAO with the telescope for observations. In this paper, we apply a high-performance correction technique, previously proposed, to rectify the static aberration in our PAO (Deqing et al. 2022). Focal plane PSF images, before and after correction, are depicted in Figure 15(a) and (b). The initial focal plane PSF exhibits significant static wave front errors. However, these errors are substantially reduced following correction by our technique, as evident in Figure 15(b).

After correcting the static aberrations, a performance evaluation is conducted for our PAO system supported by the hybrid parallel technique. Given the limitations of the first resonance of the membrane, the current DM with 88 actuators in a 10×10 configuration used in our PAO can operate at speeds around 1200 Hz. To fully assess the effectiveness of the hybrid parallel technique in our PAO, we utilize a 20×20 configuration (316 effective sub-apertures) for wave front sampling. Due to the support of the hybrid parallel programming technique, our PAO system achieves speeds of around 1100 Hz. In contrast, the PAO driven by the single LabVIEW program can just get a speed of around 600 Hz, which means the hybrid parallel technique is able to increase the PAO's running speed by nearly 76%. In our tests, the PAO initially operates with stable correction to obtain a perfect PSF image, as all static aberrations have been previously corrected. Figure 16 showcases the PSF with an SR of 0.99, stably

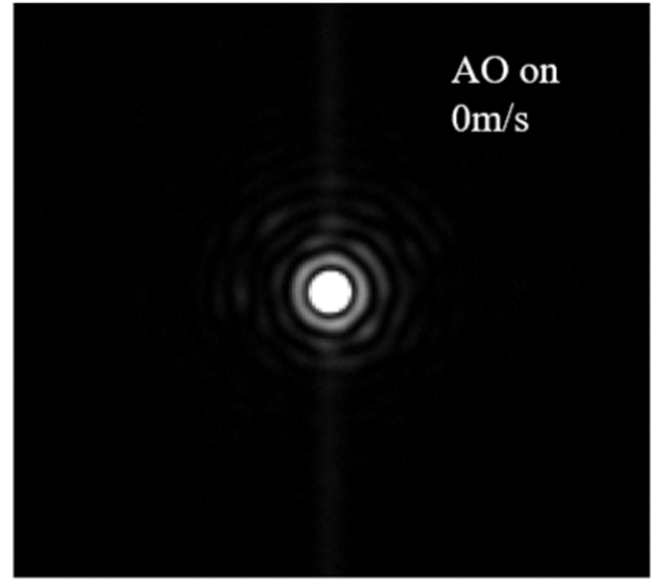


Figure 16. Focal plane PSF image: wind speed is 0 m s^{-1} without applying phase screen aberration.

corrected by the PAO with no aberrations present. Subsequently, the phase screen is subjected to wind speeds of 5 m s^{-1} , 15 m s^{-1} , and 25 m s^{-1} , to simulate atmospheric turbulence using our simulation platform. The results, shown in Figure 17 from left to right respectively, display the focal-plane images with and without the PAO's closed-loop correction. By analyzing 100 focal-plane PSF images under different wind speed conditions, we calculate the SR to reduce measurement errors. Figure 18 presents the final measured SR values achieved by our PAO's closed-loop correction at different wind speeds, under an averaged seeing condition of $r_0 = 13 \text{ cm}$. It is evident that higher wind speeds diminish focal-plane imaging quality. Even with wind speeds adjusted to 25 m s^{-1} , our PAO system, supported by the hybrid parallel technique, continues to provide stable closed-loop correction, albeit with a reduced

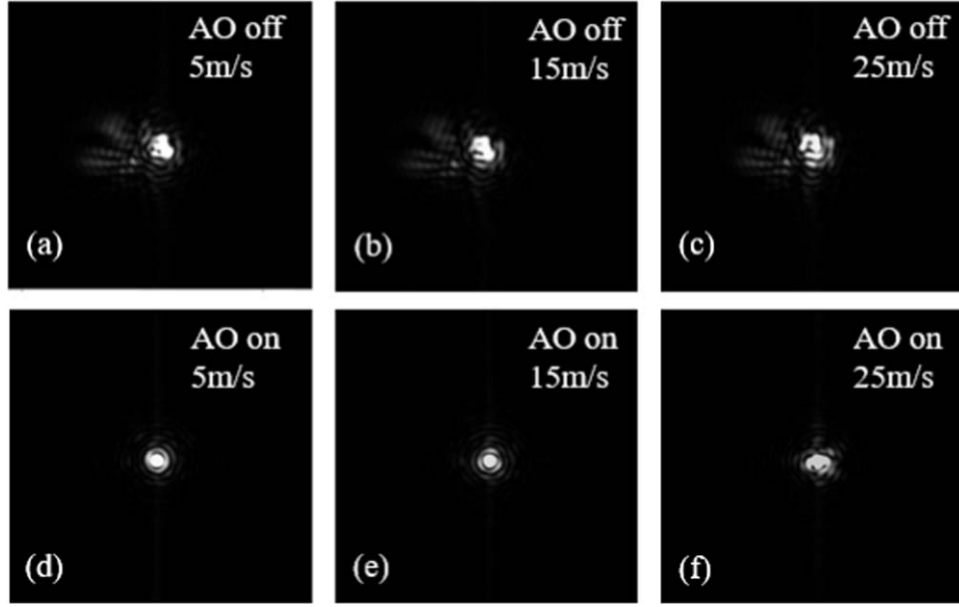


Figure 17. Focal plane PSF images without (a)–(c) and with (d)–(f) the PAO closed loop correction under the wind speed of 5 m s^{-1} , 15 m s^{-1} and 25 m s^{-1} , from left to right respectively.

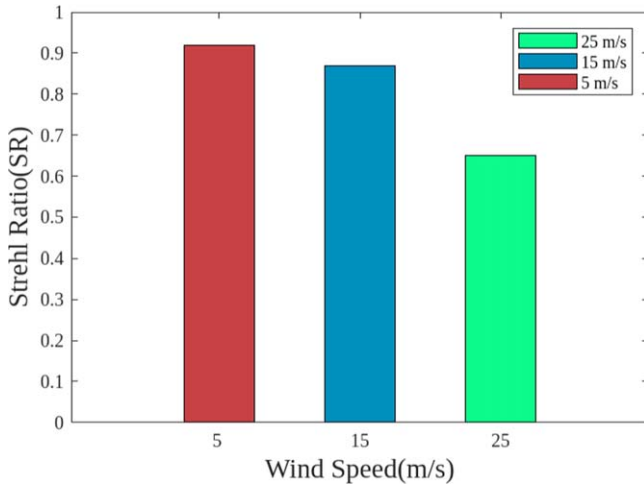


Figure 18. AO performance measurement results at different wind speeds.

7. Conclusion

In this publication, we have introduced the first dedicated Th-Net for adaptive threshold combined with the LabVIEW-based hybrid programming technique. This innovative approach effectively addresses the limitations inherent in LabVIEW, enabling multi-processing parallel calculations for centroid module-based methods in nighttime AO wave front sensing and dramatically improving accuracy of wave front detection. The hybrid parallel calculation code, crafted in C++ and Python, seamlessly integrates into LabVIEW as a user-defined function. When compared to the native LabVIEW centroid calculations, our PAO system's operational speed, powered by the hybrid parallel code, exhibits an increase of 71%–114% across various sub-aperture configurations. At the same time, the PAO's SR is improved by 5.7%–22.6% at different observational wind speeds. This enhancement places its performance on par with other state-of-the-art nighttime AO systems. To evaluate the efficacy of our Th-Net and hybrid parallel programming PAO system, we developed a high-speed atmospheric turbulence simulator platform, a first of its kind dedicated to our PAO system. This platform enables the simulation of various seeing conditions and wind speeds, providing a robust means to evaluate our PAO's performance by measuring the final SR. Thanks to the comprehensive utilization of available multi-core calculations, this Th-Net and hybrid parallel technique not only accelerates centroid measurement but also proves more efficient than single LabVIEW programming. Additionally, the hybrid approach retains the unique advantages of LabVIEW in AO control software

SR of 0.65. Under the same experimental conditions, using a uniform threshold driven by the single LabVIEW case, the SR values at wind speeds of 5 m s^{-1} , 15 m s^{-1} , and 25 m s^{-1} are 0.87, 0.76, and 0.53, respectively. In contrast, our hybrid parallel technique, which combines Th-Net, improves the SR by 5.7%, 14.5%, and 22.6% at the three different wind speeds, respectively. In our experiments, the PAO operates reliably at around 1100 Hz, aligning with our previous evaluation results listed in Table 6. These test results unequivocally demonstrate that our hybrid parallel technique significantly enhances the PAO's operational speed while maintaining correction stability.

development, eliminating the need for writing complex communication protocols and significantly reducing development time. In conclusion, our Th-Net and hybrid parallel technique mark a significant stride toward realizing a low-cost, reliable, and quickly replicable PAO system for future large telescopes.

References

- Bershad, B. N., Zekauskas, M. J., & Midway, J. 1991, Shared Memory Parallel Programming with Entry Consistency for Distributed Memory Multiprocessors, Technical Report CMU-CS-91-170, Carnegie-Mellon Univ.
- Conan, R., & Correia, C. 2014, *Proc. SPIE*, **9148**, 91486C
- Currie, T., Brandt, G. M., Brandt, T. D., et al. 2023, *Sci*, **380**, 198
- Deqing, R., Tianyu, Z., & Gang, W. 2022, *OEE*, **49**, 210319
- DuBose, T. B., Gardner, D. F., & Watnik, A. T. 2020, *OptL*, **45**, 1699
- Gropp, W. D. 2000, in European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting (Berlin: Springer), 160
- Gulli, A., Kapoor, A., & Pal, S. 2019, Deep Learning with TensorFlow 2 and Keras: regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API (Birmingham: Packt Publishing)
- Guo, Y., Wu, Y., Li, Y., Rao, X., & Rao, C. 2022, *MNRAS*, **510**, 4347
- Jia, P., Ma, M., Cai, D., et al. 2021, *MNRAS*, **503**, 3194
- Jin, H., Jespersen, D., Mehrotra, P., et al. 2011, *ParC*, **37**, 562
- Kahanamoku, S., Alegado, R., Kagawa-Viviani, A., et al. 2020, arXiv:2001.00970
- Keller, C. U., Plymate, C., & Ammons, S. M. 2003, *Proc. SPIE*, **4853**, 351
- Lee, J. H., Shin, S., Park, G. N., Rhee, H.-G., & Yang, H.-S. 2017, *COPP*, **1**, 107
- Lee, S., Benaissa, A., Djenidi, L., Lavoie, P., & Antonia, R. 2012, *PhFI*, **24**, 075101
- Marois, C., Zuckerman, B., Konopacky, Q. M., Macintosh, B., & Barman, T. 2010, *Natur*, **468**, 1080
- Ramsay, S., Amico, P., Bezawada, N., et al. 2020, *Proc. SPIE*, **11203**, 1120303
- Rao, C., Zhu, L., Rao, X., et al. 2010, *ChOpL*, **8**, 966
- Ren, D., Penn, M., Wang, H., Chapman, G., & Plymate, C. 2009, *Proc. SPIE*, **7438**, 74380P
- Ren, D., & Wang, G. 2020, *PASJ*, **72**, 30
- Ren, D.-Q., Zhang, T.-Y., & Wang, G. 2021, *RAA*, **21**, 181
- Rimmele, T. R. 2004, *Proc. SPIE*, **5490**, 34
- Roddier, N. A. 1990, *OptE*, **29**, 1174
- Schmidt, D., Rimmele, T., Marino, J., et al. 2019, in Proc. AO4ELT6
- Smith, L., & Bull, M. 2001, *ScPro*, **9**, 83
- Swanson, R., Lamb, M., Correia, C., Sivanandam, S., & Kutulakos, K. 2018, *Proc. SPIE*, **10703**, 107031F
- Tyson, R. K., & Frazier, B. W. 2022, Principles of Adaptive Optics (Boca Raton, FL: CRC Press)
- Zhu, Y.-T., Dou, J.-P., Zhang, X., et al. 2021, *RAA*, **21**, 082